UNIVERSIDAD CIENTÍFICA DEL SUR

FACULTAD DE CIENCIAS EMPRESARIALES

CARRERA PROFESIONAL DE INGENIERÍA DE SISTEMAS DE INFORMACIÓN Y GESTIÓN

Modelo de Integración entre Arquitecturas de Microservicios (MSA) y dispositivos de Internet de las Cosas (IOT), para la eficiencia operativa de una empresa de telecomunicaciones en el Perú

Trabajo de investigación para optar el grado académico de Bachiller en Ingeniería de Sistemas de Información y Gestión

Presentado por:

Edgar Antogionni Reyes Castillo

LIMA- PERÚ

2019

CIENTÍFICA UNIVERSIDAD CIENTÍFICA DEL SUR
MEJORAS TÚ, MEJORA EL MUNDO

ANEXO 6

RESULTADO DEL INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN PARA OPTAR EL GRADO DE BACHILLER

Fecha: 18/12/2019

Facultad	Ciencias Empresariales

Carreras de pre grado

Ingeniería de Sistemas Empresariales		Ingeniería de Sistemas de Información y Gestión - CPE	X
Ingeniería Económica y de Negocios		Administración de Empresas - CPE	
Administración de Negocios Internacionales			
Administración de Empresas			
Marketing y Administración			

Semestre /ciclo académico /módulo :

Código	Alumno	Correo electrónico	Teléfono	Nota*
100007402	Reyes Castillo Edgar Antogionni	100007402@ucientifica.edu.pe	987334131	17

*Nota final del curso aprobatoria

Título del trabajo de investigación:

Modelo de Integración entre Arquitecturas de Microservicios (MSA) y dispositivos de Internet de las Cosas (IOT), para la eficiencia operativa de una empresa de telecomunicaciones en el Perú.

El Jurado revisor toma en cuenta estos criterios de evaluación para calificar el **Trabajo de Investigación** para optar el Grado de Bachiller:

Aprobado (13 – 15)
Aprobado – Muy Buena (16 – 18)
Aprobado – Sobresaliente (19 – 20)
Desaprobado (0 – 12)

Comentario/sugerencias/recomendaciones/observaciones:

BUEN TRABAJO , SOLAMENTE FALTA MEJORAR EL FORMATO APA

Atentamente,

Profesor del curso
Mba Deisy Lizbeth Acosta Ticse

*Este documento puede ser enviado por correo electrónico

INDICE DE CONTENIDO

Índice de tablas

Índice de figuras

INTRODUCCION

En muchas de empresas, en el área de TI, siempre se ha tratado de encontrar mejores formas de desarrollar sistemas de información que permitan mejorar las operaciones en los procesos de negocio. Algunas empresas vienen adaptándose a los cambios tecnológicos, adoptando nuevas tecnologías que ofrece el mercado. Otras han continuado con sus propias soluciones, que les ha permitido tener éxito por muchos años, manteniendo estas soluciones como son los conocidos sistemas legados. Estas aplicaciones legados han operado muy bien durante muchos años y se han acoplando a los cambios tecnológicos en las empresas. Sin embargo, las empresas evolucionan y crecen, tanto en operatividad como en información, y muchas de estas soluciones quedan en obsolescencia.

Cuando se menciona a los sistemas legados (legacy) en las soluciones de TI, se relacionan con los sistemas monolíticos, que son sistemas heredados y que mantienen una arquitectura monolítica. Esto quiere decir, que son estructura de software que contienen toda la lógica del negocio empaquetados en un solo componente; donde las operaciones de los negocios cada vez se ven más afectadas por el bajo rendimiento que estos componentes genera y por el incremento de la demanda de operaciones que exige el propio negocio. El plan de la empresa es aumentar sus ingresos y por consiguiente, soportar el aumento de las operaciones.

Los equipos de proyecto de TI en las empresas han venido realizando esfuerzos, adoptando nuevos retos y paradigmas en las soluciones de sus sistemas de información. Con la llegada de la plataforma de servicios en la nube, algunos de estos equipos han tratado de llevar sus soluciones a esta plataforma, pero los servicios en la nube es la red de internet y eso no ha sido nada fácil de migrar, ya que sus soluciones de software tienen mucho código y tiene mucha funcionalidad. Sin embargo, es viable mover parte del software que realiza una cierta funcionalidad del negocio, que es la que se tiene ganada dentro de la organización, y es cuando se piensa en desacoplar y segmentar ciertos componentes del software para garantizar el buen funcionamiento de la operación del negocio, es allí donde entra el contexto de Microservicios.

Una arquitectura de Microservicios permite granular o segmentar el software en pequeños componentes, con una funcionalidad particular e independiente, que realiza una operación específica dentro del negocio. Para una empresa, como es el sector de telecomunicaciones que realiza millones de transacciones por milisegundo, este arquitectura es el más conveniente.

Con la creciente demanda de dispositivos móviles como los Smartphone y otros dispositivos para hogares dentro del contexto de internet de las coas (IoT), necesitan que la respuesta de un servicio que ofrece el negocio sea más rápida y en tiempo real. Por ello, estos tipos de empresas necesitan cambiar su infraestructura tecnológica (hardware, software y comunicaciones) a una arquitectura que les permita ofrecer un servicio más eficiente y dar soporte a los cambios que demandan los usuarios, de un manera más fácil y dinámica sin afectar la continuidad del servicio.

CAPITULO I: DESCRIPCION DEL PROBLEMA

1.1. Descripción de la realidad problemática

A continuación, se mencionan sustentos respecto a las arquitecturas monolíticas, como por ejemplo:

El monolito crece con el tiempo. Adquiere nuevas funcionalidades y líneas de código a un ritmo alarmante. En poco tiempo se convierte en una presencia gigante grande y aterradora en nuestra organización que las personas tienen miedo de tocar o cambiar. ¡Pero no todo está perdido! Con las herramientas adecuadas a nuestra disposición, podemos matar a esta bestia (Newman, 2015, pág. 79).

Asimismo, existe otro autor que explica un similar contexto:

Las aplicaciones monolíticas pueden ser exitosas, pero cada vez más personas sienten frustraciones con ellas, especialmente a medida que se implementan más aplicaciones en la nube. Los ciclos de cambio están unidos: un cambio realizado en una pequeña parte de la aplicación requiere que todo el monolito sea reconstruido y desplegado. Con el tiempo, a menudo es difícil mantener una buena estructura modular, por lo que es más difícil mantener los cambios que solo deberían afectar a un módulo dentro de ese módulo. El escalado requiere el escalado de toda la aplicación en lugar de partes que requieren un mayor recurso (Lewis & Fowler, 2014).

Cuando hablamos de microservicios, una pregunta común es si esto es solo Arquitectura Orientada a Servicios (SOA) que vimos hace una década. Hay mérito en este punto, porque el estilo de microservicio es muy similar al que algunos defensores de SOA han estado a favor. El problema, sin embargo, es que SOA significa demasiadas cosas diferentes, y que la mayoría de las veces nos encontramos con algo llamado "SOA" es significativamente diferente al estilo que estamos describiendo aquí, generalmente debido a un enfoque en los ESB utilizados para integrar aplicaciones monolíticas (Lewis & Fowler, 2014).

¿Cuál es la relación que existe entre la Arquitectura de Microservicios y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú?

Problemas Específicos.-

1. ¿Cuál es la relación que existe, entre la Arquitectura tecnológica, como parte de la Arquitectura de Microservicios (MSA), y la eficiencia operativa

en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú?

2. ¿Cuál es la relación que existe, entre la Arquitectura REST como estandarización en la Arquitectura de Microservicios (MSA), y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú?

2.1. Objetivos del proyecto

2.1.1. Objetivo general

El objetivo de este trabajo de investigación, es determinar la relación que existe entre Arquitectura de Microservicios y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú.

Para tal fin se implementará el sistema de información para dispositivos de la industria automotriz.

2.1.2. Objetivos específicos

- Determinar la relación que existe, entre la Arquitectura tecnológica como parte de la Arquitectura de Microservicios (MSA), y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú.

- Determinar la relación que existe, entre la Arquitectura REST como estandarización en la Arquitectura de Microservicios (MSA), y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú.

2.2. Justificación del proyecto

A continuación, se menciona el sustento la adopción de una arquitectura de Microservicios:

Muchas organizaciones han descubierto que al adoptar arquitecturas de microservicio de grano fino, pueden entregar software más rápido y adoptar tecnologías más nuevas. Los microservicios nos dan significativamente más libertad para reaccionar y tomar decisiones diferentes, lo que nos permite responder más rápido al cambio inevitable que nos afecta a todos (Newman, 2015, pág. 1).

Existe otro autor que explica un similar contexto:

La comunidad de microservicios favorece un enfoque alternativo: puntos finales inteligentes y tuberías tontas. Las aplicaciones creadas a partir de microservicios pretenden ser lo más desacopladas y cohesivas posible: poseen su propia lógica de dominio y actúan más como filtros en el sentido clásico de Unix: reciben una solicitud, aplican la lógica según corresponda y producen una respuesta (Lewis & Fowler, 2014).

Asimismo, existe otro autor que resume un similar contexto:

El presente proyecto de tesis tuvo por objetivo migrar los servicios de compra de paquetes, pago de recibo y recarga de saldo a una arquitectura basada en microservicios con el fin de soportar las transacciones realizadas por el aplicativo móvil en una empresa de telecomunicaciones sobre Amazon Web Services. La metodología utilizada fue SCRUM para la gestión del producto y la implementación de las buenas prácticas DevOps para mejorar la coordinación y comunicación entre las áreas de Desarrollo, Operaciones y Calidad con el fin de alinear los equipos de trabajo para la implementación de esta nueva arquitectura. Como resultado, se logró optimizar el tiempo de respuesta de los servicios de compra de paquetes, pago de recibo y recarga de saldo, asimismo se minimizó el porcentaje de incidencias de estos servicios y, por último, se incrementó el número de transacciones exitosas realizadas por el aplicativo móvil. Finalmente se concluye que al implementar la arquitectura basada en microservicios logra soportar el constante incremento de la demanda de transacciones realizadas mediante el aplicativo móvil (Pereira & Rumiche 2017).

CAPITULO II: MARCO TEORICO

3.1. Antecedentes

López y Maya (2017) Arquitectura de Software basada en Microservicios para Desarrollo de Aplicaciones Web. Resumen:

> Actualmente, el proceso de desarrollo de software que realiza la Coordinación General de Tecnologías de la Información y Comunicación (CGTIC) de la Asamblea Nacional del Ecuador (ANE) constituye el empleo de una arquitectura de software tradicional o monolítica que ha sido adoptada del lenguaje de programación utilizado, la plataforma o de la experiencia del personal del área de desarrollo; por el aspecto monolítico, este tipo de aplicaciones empaquetan toda la funcionalidad en una sola y gran unidad ejecutable (un solo archivo o aplicación), lo que ha provocado dificultades en aspectos como mantenimiento, escalabilidad y entregas. El objetivo del presente estudio fue identificar las tecnologías, metodología y arquitectura que utiliza la CGTIC para el desarrollo de aplicaciones web y la correspondiente identificación de las tecnologías existentes para el desarrollo e implementación de microservicios, utilizando como base de la investigación un enfoque cualitativo, con un tipo de investigación descriptiva y diseño documental. Se empleó la técnica de grupo focal aplicado a los funcionarios del área de desarrollo de software de la CGTIC, revisión bibliográfica de arquitectura de microservicios. Como avance de la investigación, el análisis ha permitido identificar el estado del arte respecto a microservicios y su implementación así como la identificación de los requisitos y necesidades relativos al desarrollo de aplicaciones web y como satisfacerlas mediante el diseño de una arquitectura de software.

Arévalo del Río (2017) El enfoque de microservicios como estrategia para mejorar la calidad del software. Resumen:

> Al superar cierto tamaño, las aplicaciones se enfrentan a problemas de arquitectura que complican y demoran el desarrollo. El enfoque de microservicios es un patrón de diseño de software que aprovecha los protocolos de redes para comunicar pequeñas aplicaciones que cumplen un objetivo específico. Ésta arquitectura pretende resolver algunas de las falencias de las Arquitecturas

Orientadas a Servicios (SOA), y ofrece una interfaz simple basada en Identificadores de recursos uniformes (URIs), y la metodología REST. Las técnicas propuestas se aplicaron a una aplicación monolítica, lo que permitió el uso de tecnologías más modernas y específicas. Se analizaron los pro y contra de este enfoque, y los beneficios en calidad, velocidad de desarrollo y desacoplamiento se comparan con la versión anterior. Se encontró que este enfoque permite realizar cambios de manera controlada, disminuyendo la cantidad de fallas y aumentando la velocidad de desarrollo.

Cayo (2018) EH-UNMSM: E-Health Cloud para la mejora de procesos de la clínica universitaria UNMSM mediante una arquitectura de microservices. Resumen:

Propone el desarrollo de un E-Health Cloud describiendo el análisis de procesos, identificación de requerimientos de software, diseño de wireframes, comunicación de componentes, infraestructura, estrategia de pruebas y despliegue en la nube. El E-Health Cloud propuesto permite mejorar el proceso de diagnóstico, interoperabilidad entre áreas médicas y oportunidad de consulta en historias clínicas usando una arquitectura de software basada en microservicios. Para la gestión del presente trabajo se aplica el marco de trabajo Scrum, con el objetivo de construir un software esperado por los usuarios de la clínica.

Arboleda (2017) Propuesta metodológica para migración de sistemas web con arquitectura monolítica hacia una arquitectura basada en microservicios. Resumen:

La Ingeniería de Software ha buscado desde sus inicios el diseño, construcción y mantenimiento de aplicaciones informáticas, acorde a las necesidades organizaciones y requerimientos técnicos dados por un consumidor. Así pues, nace la Arquitectura Basada en Microservicios como un enfoque para el mejoramiento de arquitecturas tradicionales, centrada en la estabilidad, confiabilidad, alto grado de resiliencia y fácil escalamiento. Este proyecto plantea el desarrollo de una propuesta metodológica que permita facilitar la migración hacia microservicios enfocada en la descomposición de una arquitectura monolítica tradicional. Para esto se realiza un análisis de los patrones de descomposición existentes y posteriormente la identificación de los componentes

de la arquitectura basada en microservicios. Esto se complementa con la elaboración del primer prototipo “Scheme for Migration towards Microservices” (SMMicro), que, si bien no existe un proceso mecánico a seguir para una migración óptima, SMMicro permitirá establecer un punto de referencia al identificar directrices y lineamientos que faciliten una transición de arquitecturas. Finalmente, se valida sobre un estudio de caso seleccionado de la “Dirección de Gestión de Información y Procesos” (DGIP) de la Escuela Politécnica Nacional del Ecuador.

Velepucha, Flores, Torres (2019) MOMMIV: Modelo para descomposición de una arquitectura monolítica hacia una arquitectura de microservicios bajo el Principio de Ocultación de Información. Resumen:

Los cambios tecnológicos actuales y la necesidad de las empresas de migrar sus aplicaciones legadas hacia plataformas modernas han permitido plantear nuevas formas de migrar aplicaciones hacia otras arquitecturas como son los microservicios. Sin embargo, hay desafíos que deben considerarse en el proceso de migración, como son la complejidad de extraer la lógica de negocio embebida en las aplicaciones, las dificultades por cambios en las bases de datos relacionales de la aplicación que afectan la transaccionalidad de la aplicación, la creación de microservicios duplicados o que realicen más de una funcionalidad de negocio. Hasta la actualidad, no se ha encontrado en la literatura un modelo que permita tener un fundamento teórico que solvente dichos problemas. Este trabajo de investigación recoge aspectos importantes de la descomposición y la migración a través de una revisión de literatura, de donde nace y se expone un modelo que permita descomponer arquitecturas monolíticas a microservicios basado en el Principio de Ocultación de Información.

Saransig (2018) Análisis de rendimiento entre una arquitectura monolítica y una arquitectura de microservicios – tecnología basada en contenedores

La evolución tecnológica, apunta a ser más eficiente en el uso de los recursos. La producción de Software, con el tiempo ha manejado diferentes tipos de arquitecturas con el fin de que cada producto cumpla los objetivos funcionales y sea eficiente en el uso de los recursos. Este es el caso de la Arquitectura Monolítica, muy reconocida en la producción de Software, su fusión con las

Máquinas Virtuales, la ha convertido en una fórmula exitosa y efectiva para proyectos pequeños y de gran escala. La innovación ha dado lugar a nuevas arquitecturas que proponen óptimas soluciones para mejorar el proceso de producción de Software. La Arquitectura de Microservicios, va ganando terreno e incuestionablemente será parte en la toma de decisiones de los DevOps para futuros proyectos por las ventajas que esta presenta. La tecnología de Contenedores es aún poco conocida en nuestro entorno, sin embargo, las tendencias muestran que hay más acogida por esta tecnología que brinda un manejo más eficiente de los recursos en comparación a las Máquinas Virtuales. En esta investigación, se hace un análisis comparativo de rendimiento entre una Aplicación con Arquitectura Monolítica ejecutándose sobre una Máquina Virtual contra la misma Aplicación, pero esta vez basada en una Arquitectura de Microservicios y usando Contenedores, ambas combinaciones corriendo sobre un equipo con las mismas características. Se somete cada ambiente a pruebas de estrés y se analiza posteriormente los datos en bruto almacenados en archivos de logs, el resultado con la comparación correspondiente permite hacer una toma de decisiones directamente enfocada en el manejo eficiente de los recursos y la eficiencia de la producción de Software. Además, se exponen ventajas relacionadas a las nuevas metodologías de desarrollo que no se detallan en profundidad en la presente investigación, sin embargo, quedan abiertas para posteriores investigaciones.

Ruelas (2017) Modelo de composición de microservicios para la implementación de una aplicación web de comercio electrónico utilizando kubernetes. Resumen:

El comercio electrónico ha crecido en los últimos años y a consecuencia de ello ha recibido una mayor atención por parte de las empresas, las cuáles vienen invirtiendo en la implementación de aplicaciones web; incrementado su demanda y la existencia de usuarios concurrentes en determinados tiempos, ocasionando un mayor nivel de exigencia; que conlleva a problemas como la disponibilidad limitada, tiempo de respuesta excedida, y el rendimiento limitado al acceder a la aplicación web del comercio electrónico. Por otro lado la arquitectura de microservicios es una nueva tendencia que crece rápidamente en el mundo empresarial, sin embargo existe poca literatura de composición de microservicios. El objetivo de esta investigación ha sido proponer un modelo de composición de microservicios para la

implementación de una aplicación Web de comercio electrónico utilizando la tecnología Kubernetes, para lo cual se utilizó el modelo y notación el proceso de negocio de comercio electrónico, el diseño arquitectónico de la composición de los microservicios, la implementación de los microservicios en forma independiente la evaluación de éstos se ha realizado a través de atributos de calidad, y la validación del modelo propuesto usando pruebas de carga. Como resultado se obtuvo que la aplicación web funciona con una mejora significativa en un 104 %, en los indicadores de rendimiento, disponibilidad y tiempo de respuesta, en comparación con una aplicación web basado en el modelo monolítico. Por lo que el modelo de composición de microservicios presentado tiene un funcionamiento significativo.

Gómez, Anaya y Cano (2017) Un acercamiento a los microservicios. Resumen:

El desarrollo basado en microservicios es una tendencia emergente que surge de las necesidades de la industria de software, para mejorar la escalabilidad y flexibilidad de las aplicaciones web y que hoy en día se reconoce como un nuevo modelo de arquitectura conocido como microservicios. Generalmente, las aplicaciones web tradicionales siguen una arquitectura por capas, en las que se hace separación lógica de la solución en tres capas: la interfaz de usuario, la lógica de la aplicación y el sistema de gestión de datos; sin embargo, el despliegue de la solución se realiza como una unidad monolítica que se ejecuta en un solo espacio de direcciones, generando problemas ante la demanda de aplicaciones con servicios especializados que requieren manejo de grandes volúmenes de datos integrados con enfoques IoT y con requerimientos de procesamiento distribuido. Las arquitecturas basadas en microservicios proponen una arquitectura en la que cada funcionalidad de negocio se descompone en servicios web altamente cohesivos que pueden ser desplegados, evolucionados y escalados de manera independiente. Este enfoque trae beneficios como la flexibilidad, escalabilidad y productividad del equipo de trabajo, pero también conlleva nuevos retos que están siendo enfrentados, como la seguridad, el desempeño y la mantenibilidad de la solución. Uno de los aspectos más importantes para tener en cuenta en el desarrollo de este tipo de aplicaciones es el alineamiento organizacional a través de un enfoque de desarrollo orientado al dominio (DDD).

Cabrera y Cárdenas (2018) Metodología para la creación de aplicaciones basadas en microservicios para soluciones de internet de las cosas en ambientes de vida asistidos. Resumen:

> Los microservicios junto con tecnologías como Internet de las Cosas (Internet of Things - IoT), Cloud Computing, entre otras, han revolucionado el modelo de negocio de compañías orientadas a proporcionar servicios de asistencia, monitoreo y vigilancia en entornos de vida asistidos (Ambient Assisted Living - AAL); sin embargo, el desarrollo de aplicaciones basado en microservicios, rompe el esquema de desarrollo de software tradicional y aún más, el esquema organizacional, siendo una arquitectura que cada vez tiene mayor aceptación. Por otro lado, las metodologías ágiles de desarrollo de software, resaltan por su sencillez, tanto en su aprendizaje como en su aplicación, permitiendo la obtención de un producto de software confiable, de calidad y con un diseño que hace frente a los cambios continuos con entregas tempranas; con este estudio se motiva la adopción de metodologías desarrollo de software ágiles con arquitecturas de microservicios y se presenta una instancia de la misma para soluciones de software en entornos de IoT para AAL. Consecuentemente, este trabajo de titulación presenta MicroIoT, una metodología para la definición, creación y despliegue de microservicios basada en metodologías ágiles, para soluciones de IoT desplegadas en AAL, la cual; alineada con el enfoque organizacional DevOps abarca adecuadamente el desarrollo y las operaciones sobre microservicios. Además, se ha diseñado e implementado un sistema de software de IoT para AAL, como una instancia desarrollada con la metodología propuesta. MicroIoT ha sido evaluada empíricamente mediante un cuasi-experimento realizado con profesionales y estudiantes de la carrera de Ingeniería de Sistemas de la Universidad de Cuenca.

Nebel (2019) Arquitectura de microservicios para plataformas de integración. Resumen:

La integración de sistemas heterogéneos se apoya comúnmente en Plataformas de Integración (PI). Estas plataformas consisten en infraestructura especializada, que provee mecanismos para resolver incompatibilidades entre sistemas con el fin de posibilitar su comunicación. En este ámbito, el avance y la expansión de la computación en la nube [han] generado nuevos escenarios y requerimientos sobre las PIs en términos de escalabilidad y eficiencia,

entre otros. Por otro lado, la arquitectura de microservicios ha surgido recientemente impulsada por la industria, y está ganando creciente popularidad. Esta posee ventajas como el escalamiento independiente y la mantenibilidad que podrían beneficiar a las PIs en distintos escenarios. Por tanto, resulta de interés explorar las ventajas, desafíos y alternativas que introduce la arquitectura de microservicios en estas plataformas. Esta tesis estudia la aplicabilidad de la arquitectura de microservicios para la construcción de PIs. Para esto se analiza, por un lado, el impacto de utilizar dicha arquitectura en PIs, determinando escenarios en los cuales es propicia su aplicación. Por otro lado, se proponen alternativas de arquitectura y diseño para la construcción de PIs basadas en microservicios, las cuales son evaluadas en base a distintos factores. En relación al análisis de impacto, se plantea una metodología para determinar cómo es afectada una PI en base a sus atributos de calidad, obteniéndose del análisis dos resultados principales. Por un lado, se determina cómo impacta aplicar una arquitectura de microservicios en la calidad de la plataforma. Por otro, se determina un conjunto de características de los escenarios de integración (p. ej. cantidad alta de sistemas a integrar), que permite identificar si un escenario se beneficia al utilizar una PI basada en microservicios. En relación a las propuestas de arquitectura y diseño, se presenta una propuesta principal aplicable a diversos escenarios, así como variantes aplicables a contextos específicos. La propuesta principal se evalúa en base a tres factores: i) patrones y buenas prácticas de microservicios, ii) atributos de calidad de la PI, y iii) la construcción de un prototipo. Se concluye en este trabajo que la arquitectura de microservicios es aplicable para la construcción de PIs en escenarios con determinadas características y que las propuestas de arquitectura planteadas siguiendo este enfoque son técnicamente viables.

3.2. Bases teóricas

Existe una serie de desafíos al mantener una solución de negocio monolítico orientada a servicios, donde estos servicios se encuentran acoplados en un solo bloque. Si bien existen una serie de componentes transversales (programación orientada a aspectos) el cual se tiene ciertos servicios que realiza funcionalidades comunes, como pueden ser componentes de seguridad, filtros, log, archivos de propiedades, entre otros; que permiten a la solución del negocio ser auditable, monitoreable y revisable, para ver donde se encuentra el problema. Lo que se tiene en esta solución son solo algunos componentes desacoplados, lo cual permite una agilidad en el soporte de estos componentes, y no se trabajen en la misma capa Back-End de los servicios del negocio, sino,

que se mantiene la capa Back-End única y exclusivamente a las necesidades del negocio. Es entonces cuando se ingresa al proceso de desacoplamiento.

Sin embargo, las funcionalidades críticas del negocio siguen estando dentro de componentes Back-End, es decir, todas las funcionalidades del servicio acoplados en un mismo paquete.

APIs y Microservicios

Los que las APIs logran hoy en día es desacoplar tecnologías. Se implementan reales componentes y no como anteriormente se implementaban, como eran los componentes DCOM, COM, COM+, CORBA, .Net, J2EE, etc. Todos estos componentes solo eran reutilizados dentro del mismo lenguaje, una misma plataforma y bajo la misma forma de diseñar software. Hoy en día con las APIs, finalmente se logran independizar los componentes del lenguaje y de su plataforma; logrando que sus clientes puedan comunicarse con estos componentes en un mismo protocolo, este protocolo es HTTP.

Estos servicios APIs están orientados al consumidor técnico, lo cual no es un usuario final, ni un usuario no informático, sino, que es un desarrollador de software. Por lo tanto, la manera como se van a desarrollar estas APIs, está pensado que en el otro lado, se tenga un desarrollador que va hacer uso del API. Algunos ejemplos de estos APIs son: API de Facebook, Twitter, Google Maps, SAP, entre otros. Estas APIs son amigables para el desarrollador, y habla el mismo leguaje de internet, en su mayoría REST.

Los Microservicios son estas mismas APIs, pero ahora física y lógicamente independientes. Esta forma o método de arquitectura permite una gran sencillez en términos de escalabilidad, sobre todo cuando se tiene pensado la compatibilidad con diferentes plataformas como son las aplicaciones para smartphones, web, IoT, wearables, entre otras.

La arquitectura de Microservicios son componentes independientes distribuidos dentro de un sistema de información, en donde cada componente expone una funcionalidad en particular al resto del sistema. De esa forma, los sistemas se desacoplan y se modularizan a través de estos componentes independientes. Los clientes externos o interno, inclusive los mismos componentes, van a consumir estas funcionalidades entre sí.

Según Gartner (Olliffe, Matheny 2019) en su publicación web Cómo tener éxito con la arquitectura de microservicios utilizando prácticas de DevOps. Resume:

> La arquitectura de microservicios ayuda a entregar software ágil y escalable, pero el éxito requiere más que nuevos patrones de arquitectura. Los profesionales técnicos responsables de la arquitectura de la aplicación deben liderar los cambios en los procesos, la organización, las plataformas y la gestión de datos para generar un impacto comercial.

SOAP y REST

SOAP tiene una guía de cómo desarrollar servicios, simplemente no existe. SOAP no orientó jamás a como diseñar servicios, solo era una forma de implementar y desarrollar servicios, y como estos servicios iban a conversar entre sí, pero no orientó a como diseñarlos. Es entonces donde llega REST, lo cual es relativamente simple, en donde realmente te dice como diseñar servicios y esa es la razón por lo que muchos proyectos orientados a servicios están implementados en REST. La especificación REST es una representación más simple que SOAP, por lo que eliminó todo lo complejo en SOAP.

La especificación SOAP viene con una serie de estándares de cómo implementar distintos tipos de protocolos y distintas formas de hacer que los servicios conversaran dentro de SOAP, pero nadie realmente ha usado todo lo que la especificación decía, porque, SOAP fue hecho tratando de resolver todos los problemas que podrían ocurrir en la comunicación entre sistemas. Sin embargo, cuando se tiene claro la manera de comunicarse de una sola manera, de dos o tres formas a lo más; SOAP era demasiado, es por eso el éxito de REST.

3.3. Definiciones conceptuales

- SOA: Según Wikipedia, es un estilo de arquitectura de TI que se apoya en la orientación a servicios.

- ESB: Según IBM, Un ESB, o bus de servicios empresariales, es un patrón mediante el cual un componente de software centralizado realiza integraciones a sistemas de back-end (y traducciones de modelos de datos, conectividad profunda, enrutamiento y solicitudes) y hace que esas integraciones y traducciones estén

disponibles como interfaces de servicio para su reutilización por nuevos aplicaciones.

- REST: Según Mozilla, El término "Transferencia de Estado Representacional" (REST) representa un conjunto de características de diseño de arquitecturas software que aportan confiabilidad, eficiencia y escalibilidad a los sistemas distribuidos. Un sistema es llamado RESTful cuando se ajusta a estas características.

- JSON: Según la organización Json define, es un formato ligero de intercambio de datos. Es fácil para los humanos leer y escribir. Es fácil para las máquinas analizar y generar. Se basa en un subconjunto del Estándar de lenguaje de programación JavaScript ECMA-262, 3.a edición, diciembre de 1999.

- Microservicios: Según la compañía REDHAT, Los microservicios representan un estilo de arquitectura y un modo de programar software. Con los microservicios, las aplicaciones se dividen en sus componentes más pequeños, y son independientes entre sí.

- IoT: Según Deloitte, La definición de IoT podría ser la agrupación e interconexión de dispositivos y objetos a través de una red (bien sea privada o Internet, la red de redes), dónde todos ellos podrían ser visibles e interaccionar.

- DevOps: Según la compañía REDHAT, DevOps describe los enfoques para agilizar los procesos con los que una idea (como una nueva función de software, una solicitud de mejora o una corrección de errores) pasa del desarrollo a la implementación, en un entorno de producción en que puede generar valor para el usuario.

- API: Según la compañía REDHAT, Una API es un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones. API significa interfaz de programación de aplicaciones.

- MSISDN: Según Wikipedia, Las siglas MSISDN hacen referencia a Mobile Station Integrated Services Digital Network, estación móvil de la Red Digital de Servicios Integrados (RDSI). El MSISDN es el equivalente móvil del RDSI. Usado como valor, el MSISDN hace referencia al Número de Suscripción RDSI de Móvil (MSIN), cuya longitud máxima es de 15 dígitos.

- SIM: Según Wikipedia, Una tarjeta SIM (acrónimo en inglés de subscriber identity module, en español módulo de identificación de

abonado) es una tarjeta inteligente desmontable usada en teléfonos móviles y módems HSPA o LTE que se conectan al dispositivo por medio de una ranura lectora o lector SIM.

- IMSI: Según Wikipedia, IMSI es el acrónimo de International Mobile Subscriber Identity (Identidad Internacional del Abonado Móvil). Es un código de identificación único para cada dispositivo de telefonía móvil, integrado en la tarjeta SIM, que permite su identificación a través de las redes GSM y UMTS.

- Firewall: Según la compañía CISCO, Un firewall es un dispositivo de seguridad de red que monitorea el tráfico de red entrante y saliente y decide si permite o bloquea el tráfico específico en función de un conjunto definido de reglas de seguridad.

- DMZ: Según la compañía Linksys, Una DMZ ayuda a las señales electrónicas a evitar la estricta seguridad del firewall y del enrutador y abre todos los puertos para una entrega más rápida de los paquetes de datos.

- Router: Según Wikipedia, es un dispositivo que permite interconectar computadoras que funcionan en el marco de una red. Su función: se encarga de establecer la ruta que destinará a cada paquete de datos dentro de una red informática.

- Clúster: Según Wikipedia, se aplica a los conjuntos o conglomerados de servidores unidos entre sí normalmente por una red de alta velocidad y que se comportan como si fuesen un único servidor.

- Balanceador: Según Wikipedia, Un Balanceador de carga fundamentalmente es un dispositivo de hardware o software que se pone al frente de un conjunto de servidores que atienden una aplicación y, tal como su nombre lo indica, asigna o balancea las solicitudes que llegan de los clientes a los servidores usando algún algoritmo (desde un simple round-robin hasta algoritmos más sofisticados).

CAPITULO III: DESARROLLO DEL PROYECTO

4.1. Arquitectura empresarial

4.1.1. Arquitectura de Información (Datos)

Los siguientes esquemas estructura la información clave del negocio con respecto al proyecto propuesto en la investigación, para desarrollar la base de información que permitirá la persistencia de las operaciones desde la plataforma cliente (empresa de automóviles) con respecto a la infraestructura tecnológica de una empresa de telecomunicaciones.

IOT

El esquema IOT relaciona entidades con la empresa de automóviles y líneas asignadas a esta empresa con respecto a sus clientes (usuarios), líneas, planes y paquetes adquiridos.

Empresa: Entidad que caracteriza la empresa de automóviles.

Empresa Cuenta: Entidad que caracteriza la relación de la empresa de automóviles con la cuenta de contrato con la empresa de telecomunicaciones.

Teléfono: Entidad que caracteriza la línea de teléfono MSISDN con respecto a las líneas a la empresa de vehículos, provisionadas por la empresa de telecomunicaciones.

Empresa Chip: Entidad que caracteriza la relación de la línea de la empresa (MSISDN) con el chip (SIM + IMSI) proporcionado por la empresa de telecomunicaciones.

Empresa Chip Historial: Entidad que caracteriza el historial de cambios de la línea (MSISDN) con relación al chip (MSISDN).

Usuario Conductor: Entidad que caracteriza al usuario final de la línea, que es el cliente de la empresa de automóviles.

Saldo actual: Entidad que caracteriza el saldo actual de paquetes adquiridos por el usuario final.

Plan: Entidad que caracteriza los planes configurados por la empresa de telecomunicaciones, para la empresa la empresa de automóviles.

Plan Paquete: Entidad que caracteriza la relación entre los paquetes

Catálogo Paquete: Entidad que caracteriza la configuración de paquetes de servicios, ofrecidos por la empresa de telecomunicaciones, para la empresa la empresa de automóviles.

Compra de Paquetes: Entidad que caracteriza las compras de paquetes conforme a un plan, y que son adquiridos por la empresa de automóviles y por el usuario final.

Activación de Paquetes: Entidad que caracteriza la activación de paquetes adquiridos por la empresa de automóviles y el usuario final.

Transacción: Entidad que caracteriza el registro de todas las transacciones realizadas por la empresa de automóviles y el usuario final, como por ejemplo: Registro de un nuevo usuario, Activación de Línea, Cambio de Chip (SIM + IMSI), Compra de paquetes, Cambio de Plan.

Transacción Detalle: Entidad que caracteriza el detalle de la transacción.

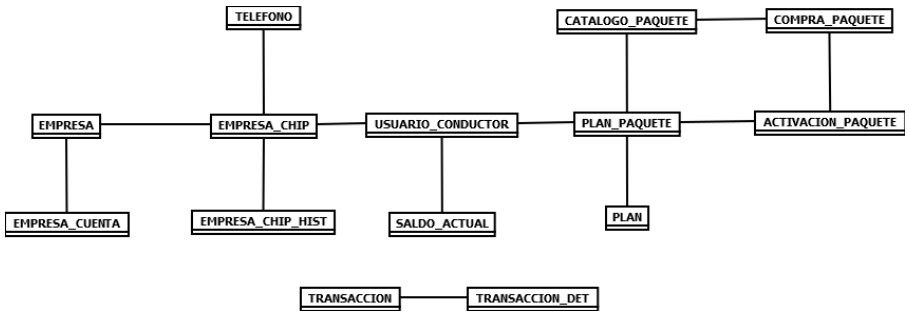

Figura N° 1: Esquema IOT
Fuente: Elaboración propia

ACTIVACIONES

El esquema Activaciones relaciona las entidades de los chips (SIM + IMSI) con relación a las activaciones y su historial de cambios. Este esquema es un enlace entre el esquema de IOT, VENTAS y FACTURACION.

Chip SimCard: Esta entidad caracteriza la información de los chips configurados por la empresa de telecomunicaciones con respecto a la empresa de automóviles.

Activación Línea: Esta entidad caracteriza las activaciones que realiza la empresa de automóviles con relación a los chips asignados.

Historial: Esta entidad caracteriza el historial y control las activaciones de la línea con relación a los chips.

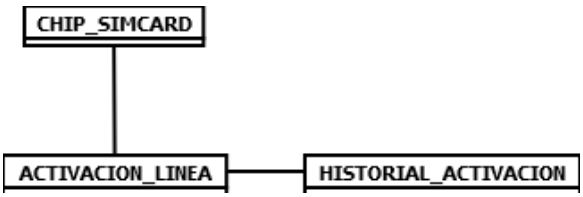

Figura N° 2: Esquema ACTIVACIONES
Fuente: Elaboración propia

VENTAS

El esquema de VENTAS relaciona las líneas activadas por la empresa de vehículos, con las ventas de las líneas provisionadas por la empresa de telecomunicaciones.

Venta: Esta entidad caracteriza la cabecera de las ventas de líneas activadas por la empresa de automóviles, provisionadas por la empresa de telecomunicaciones. Sólo se realizará la venta de una línea activada, y que fueron provisionadas por la empresa de telecomunicaciones, para la empresa de automóviles. Esto por medio de la cuenta contractual de la empresa de automóviles con el operador.

Venta Detalle: Esta entidad caracteriza el detalle de la venta, con relación a los planes vigentes y paquetes comprados.

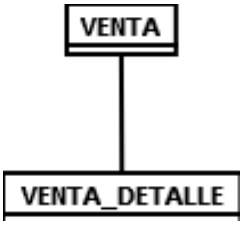

Figura N° 3: Esquema VENTAS
Fuente: Elaboración propia

FACTUACION

El esquema de FACTURACION relaciona las ventas realizadas de la línea y los paquetes de servicios conforme a los planes establecidos para la empresa de automóviles.

Factura: Esta entidad caracteriza la cabecera de facturación, con relación a la línea activada por la empresa de automóviles y usuarios finales.

Factura Detalle: Esta entidad caracteriza el detalle de la facturación, con relación a los paquetes de servicios adquiridos por la empresa de automóviles y los usuarios finales.

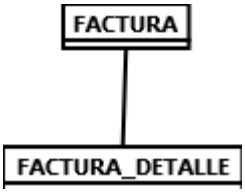

Figura N° 4: Esquema FACTURACION
Fuente: Elaboración propia

4.1.2. Arquitectura de Negocios (Procesos)

La arquitectura de negocio expone en alto nivel, las actividades que realiza el proceso negocio, propuesto para este proyecto de investigación.

Gestión de Provisión de Líneas

Es proceso gestionará las operaciones relacionadas a la líneas, que han sido asignadas a la empresa de automóviles por parte del operador, en este caso, una empresa de telecomunicaciones.

Provisionar Línea: Esta operación define las actividades dentro del proceso Provisión de Líneas, la provisión de la misma; es decir, actividades para realizar una provisión de la línea, desde la plataforma de la empresa de automóviles, hacia la infraestructura tecnológica del operador.

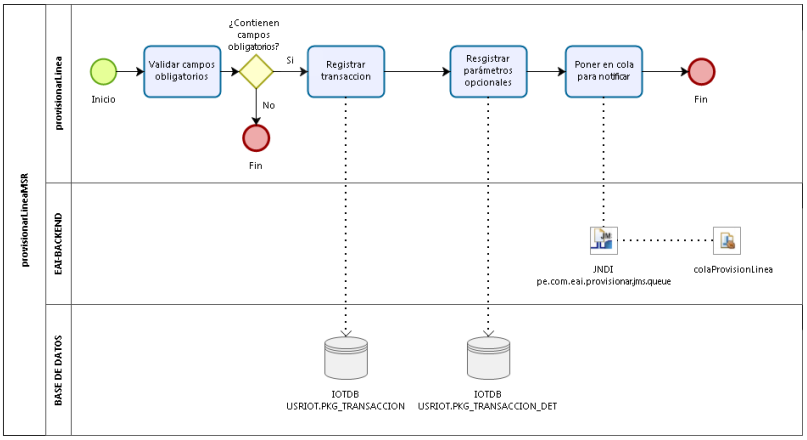

Figura N° 5: Provisión de una línea
Fuente: Elaboración propia

Consultar Información de la Línea: Esta operación define las actividades dentro del proceso Provisión de Líneas, para realizar las consultas sobre información de la línea provisionada por la plataforma de la empresa de automóviles, hacia la infraestructura tecnológica del operador.

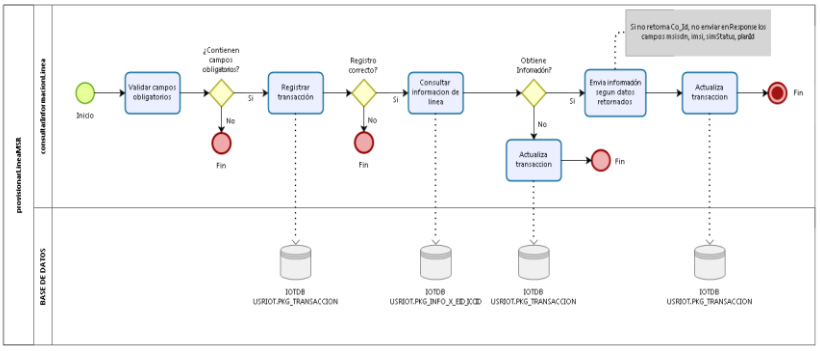

Figura N° 6: Consultar información de la línea
Fuente: Elaboración propia

Desaprovisionar Línea: Esta operación define las actividades dentro del proceso Provisión de Líneas, para realizar una desaprovisión de la línea, previamente provisionada, desde la plataforma de la empresa de automóviles, hacia el la infraestructura tecnológica del operador.

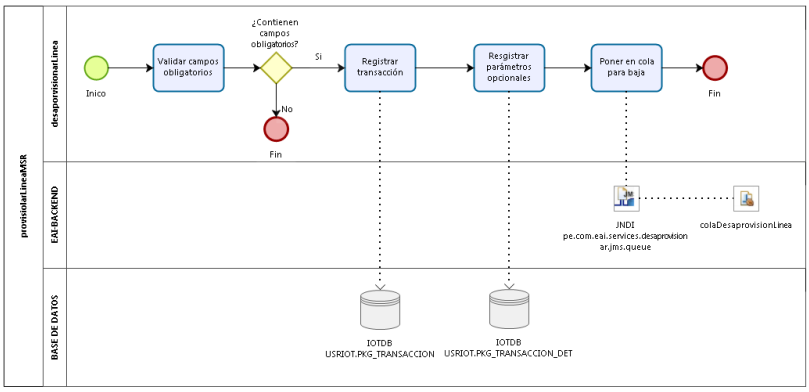

Figura N° 7: Desaprovisionar línea
Fuente: Elaboración propia

Gestión de Planes

Es proceso gestionará las operaciones relacionadas a los planes, que han sido asignadas a la empresa de vehículos por parte del operador, en este caso, una empresa de telecomunicaciones.

Consultar Planes: Esta operación define las actividades dentro del proceso Planes, para realizar la consulta de planes asignados a la empresa de automóviles; desde su plataforma, hacia la infraestructura tecnológica del operador.

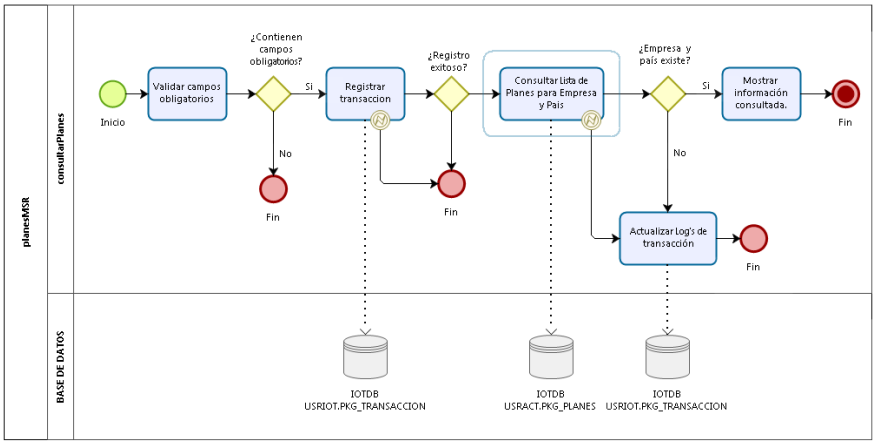

Figura N° 8: Consultar planes
Fuente: Elaboración propia

Cambiar Plan: Esta operación define las actividades dentro del proceso Planes, para realizar el cambio de plan de una línea activa; desde la plataforma de la empresa de automóviles, hacia la infraestructura tecnológica del operador.

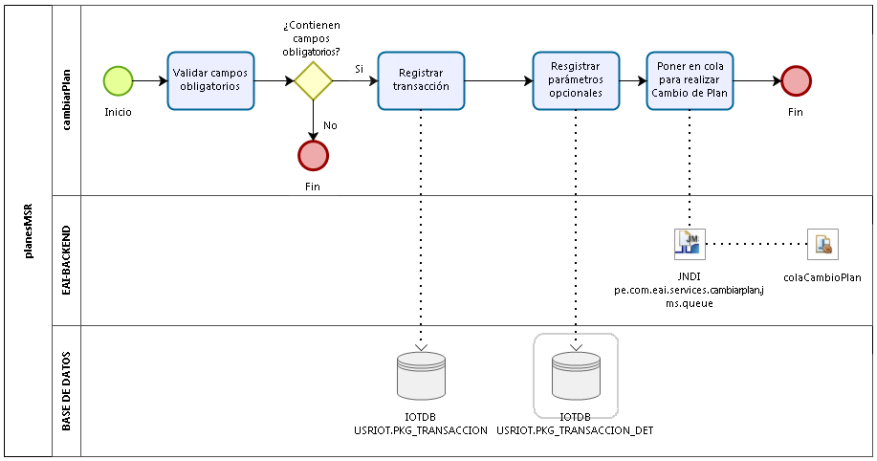

Figura N° 9: Cambiar plan
Fuente: Elaboración propia

Gestión Usuario Conductor

Es proceso gestionará las operaciones relacionadas la información del usuario conductos, quienes son los usuarios finales de los servicios, ofrecidos por el operador, en este caso, una empresa de telecomunicaciones.

Registrar Usuario Conductor: Esta operación define las actividades dentro del proceso Gestión Usuario Conductor, para registrar la información del usuario final; desde la plataforma de la empresa de automóviles, hacia la infraestructura tecnológica del operador.

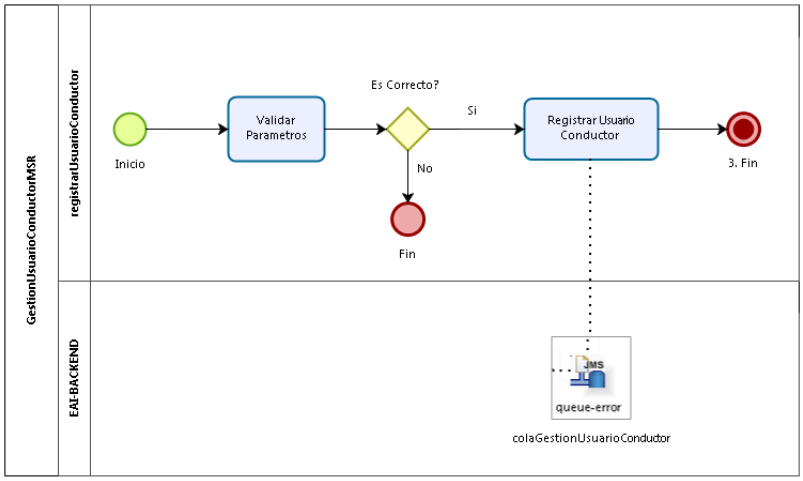

Figura N° 10: Registrar usuario conductor
Fuente: Elaboración propia

Modificar Usuario Conductor: Esta operación define las actividades dentro del proceso Gestión Usuario Conductor, para modificar la información del usuario final; desde la plataforma de la empresa de automóviles, hacia la infraestructura tecnológica del operador.

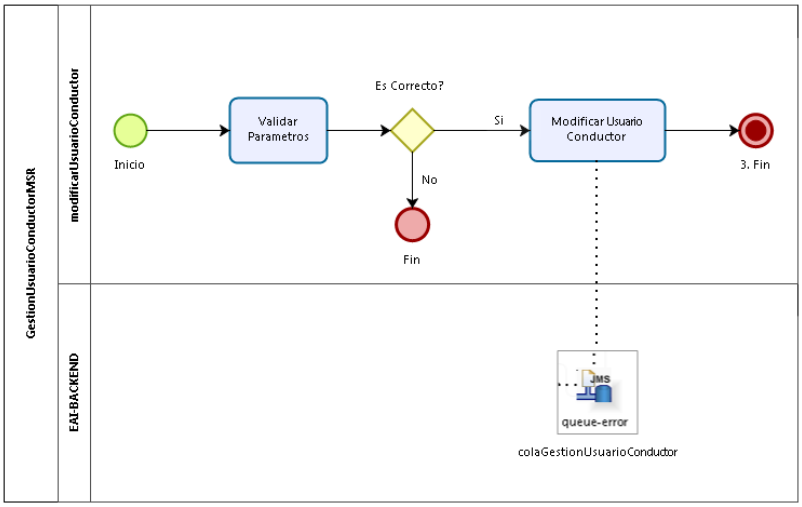

Figura N° 11: Modificar usuario conductor
Fuente: Elaboración propia

Eliminar Usuario Conductor: Esta operación define las actividades dentro del proceso Gestión Usuario Conductor, para eliminar la información del usuario final; desde la plataforma de la empresa de automóviles, hacia la infraestructura tecnológica del operador.

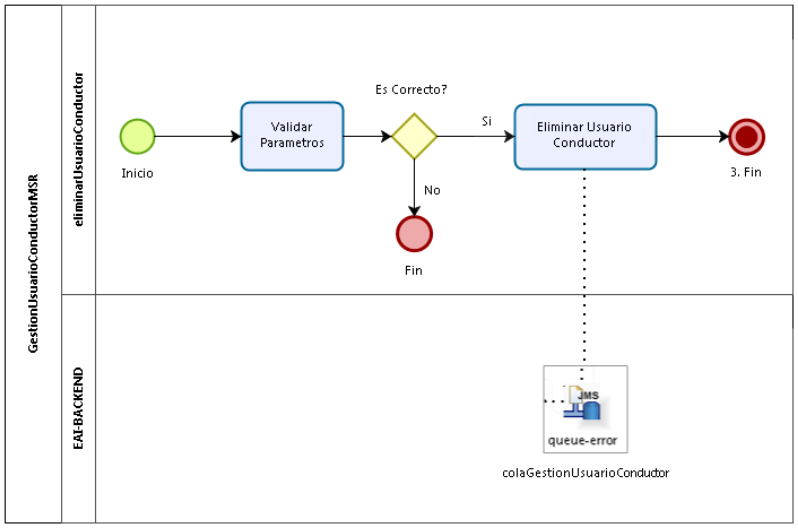

Figura N° 12: Eliminar usuario conductor
Fuente: Elaboración propia

Consultar Usuario Conductor: Esta operación define las actividades dentro del proceso Gestión Usuario Conductor, para consultar la información del usuario final; desde la plataforma de la empresa de automóviles, hacia la infraestructura tecnológica del operador.

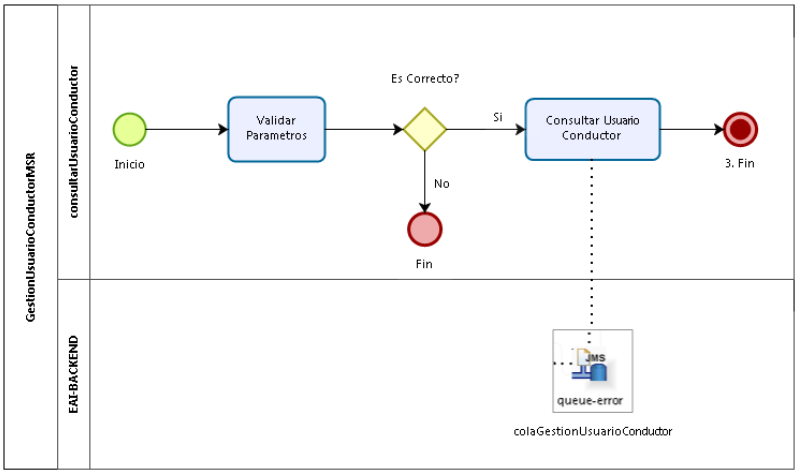

Figura N° 13: Consultar usuario conductor
Fuente: Elaboración propia

Gestión Paquete de Datos

Es proceso gestionará las operaciones relacionadas a los paquetes de datos, que son parte de los planes asignados a la empresa de automóviles por parte del operador, en este caso, una empresa de telecomunicaciones.

Comprar Paquetes: Esta operación define las actividades dentro del proceso Paquetes de Datos, para realizar la compra de paquetes por parte de la empresa de automóviles o el usuario final; desde la plataforma de la empresa de automóviles, hacia la infraestructura tecnológica del operador.

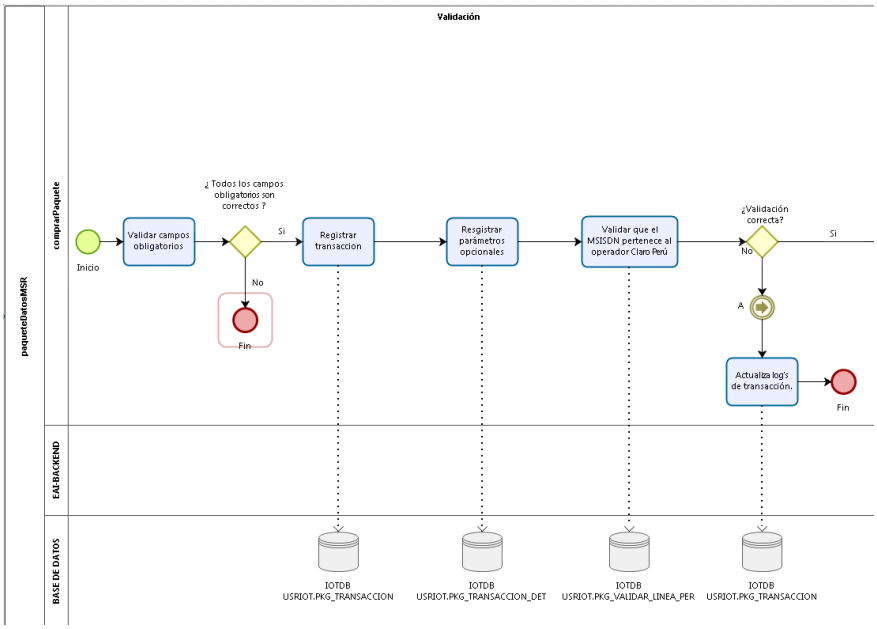

Figura N° 14: Comprar paquetes (Parte1)
Fuente: Elaboración propia

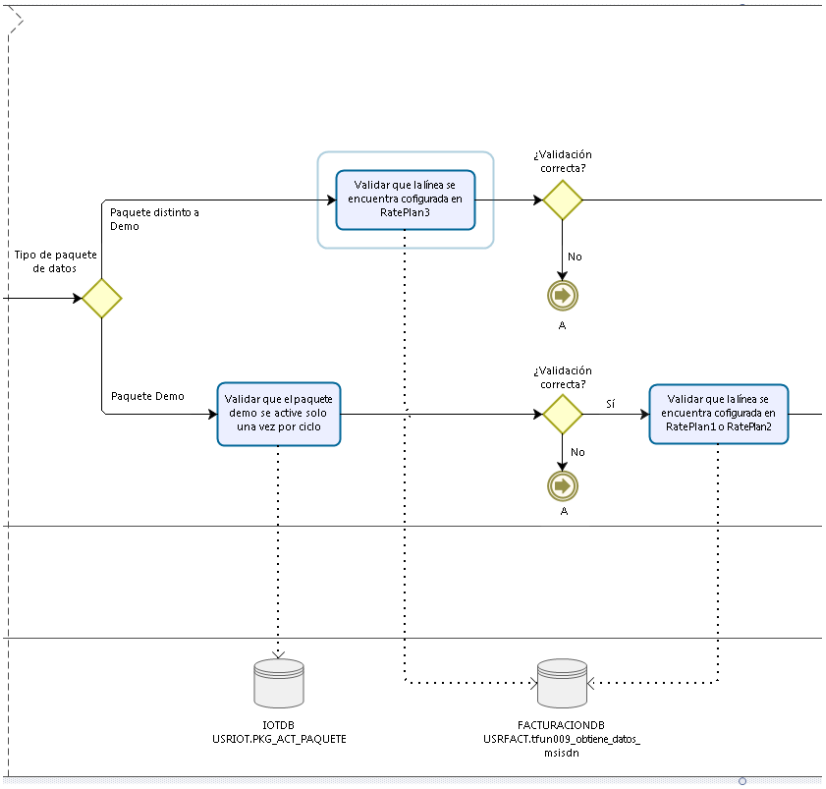

Figura N° 15: Comprar paquetes (Parte2)
Fuente: Elaboración propia

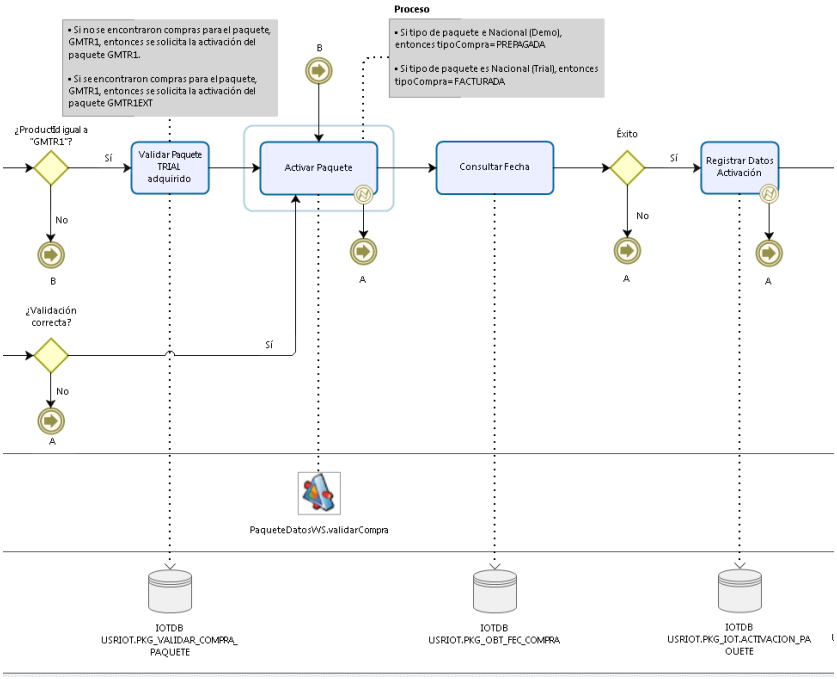

Figura N° 16: Comprar paquetes (Parte3)
Fuente: Elaboración propia

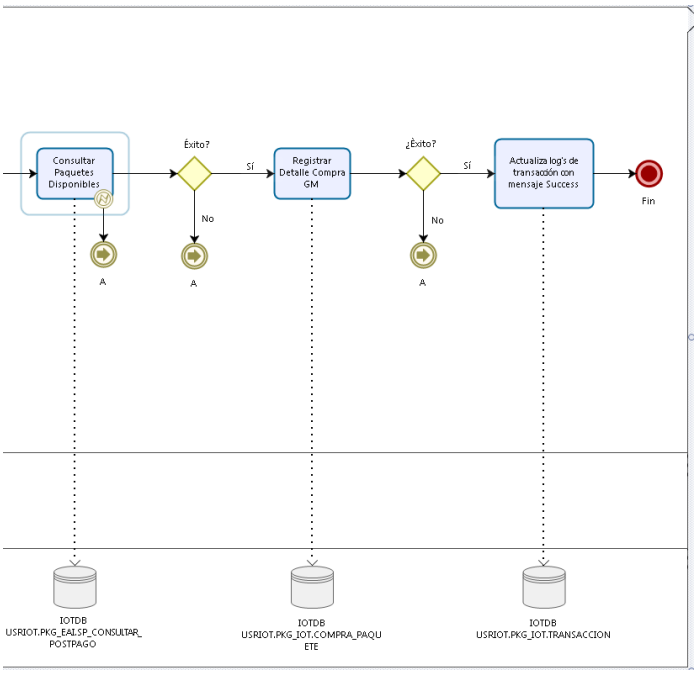

Figura N° 17: Comprar paquetes (Parte4)
Fuente: Elaboración propia

Consultar Paquetes: Esta operación define las actividades dentro del proceso Paquetes de Datos, para realizar la consulta de paquetes adquiridos, por parte de la empresa de automóviles o el usuario final; desde la plataforma de la empresa de automóviles, hacia la infraestructura tecnológica del operador.

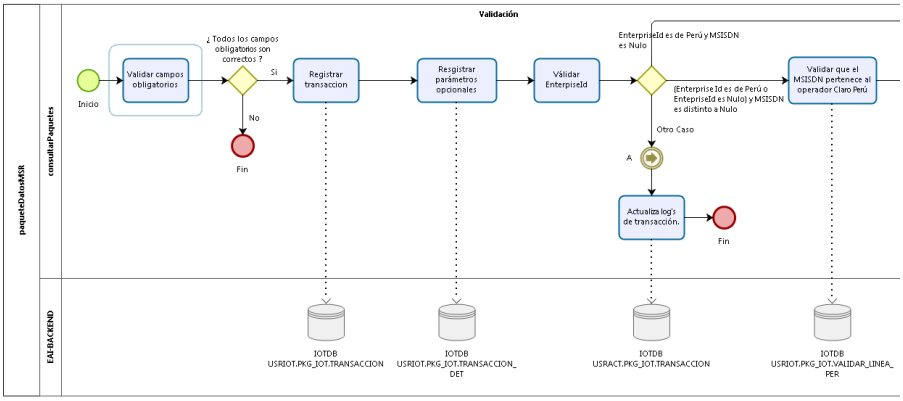

Figura N° 18: Consultar paquetes (Parte1)
Fuente: Elaboración propia

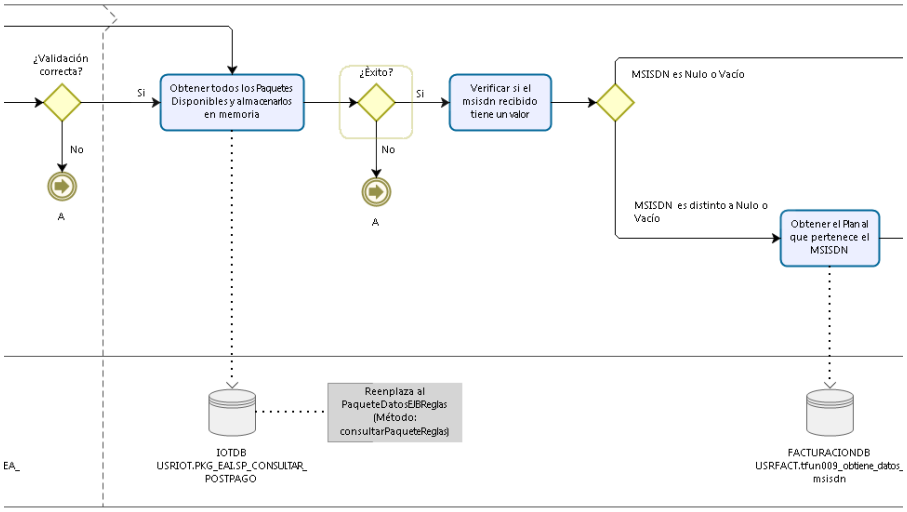

Figura N° 19: Consultar paquetes (Parte2)
Fuente: Elaboración propia

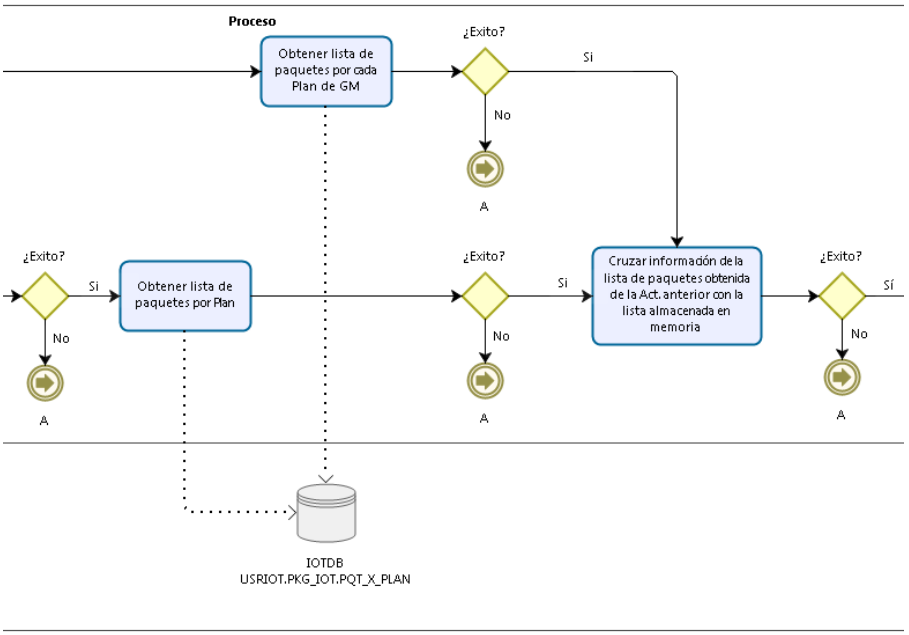

Figura N° 20: Consultar paquetes (Parte3)
Fuente: Elaboración propia

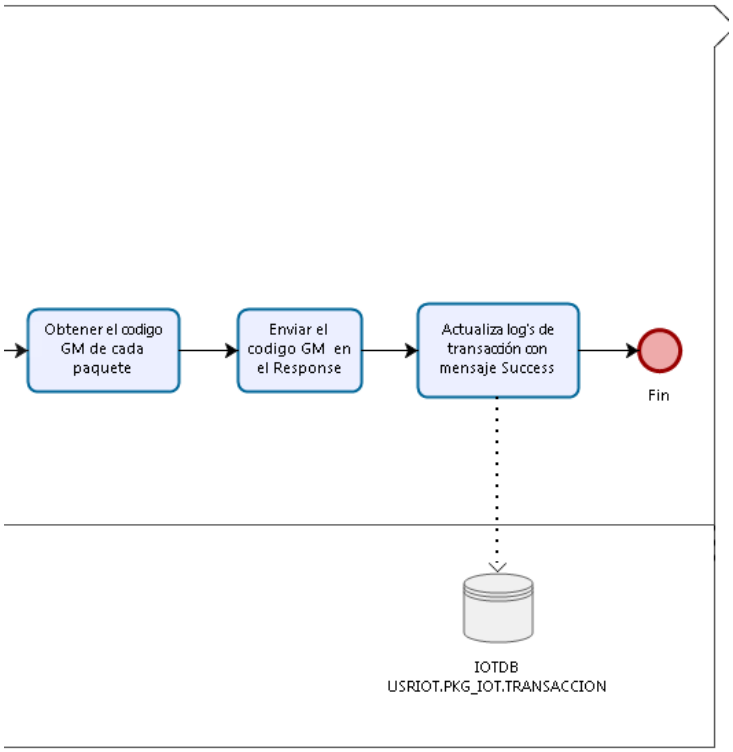

Figura N° 21: Consultar paquetes (Parte4)
Fuente: Elaboración propia

Cancelar Paquetes: Esta operación define las actividades dentro del proceso Paquetes de Datos, para realizar la cancelación de paquetes adquiridos, por parte de la empresa de automóviles o del usuario final; desde la plataforma de la empresa de automóviles, hacia la infraestructura tecnológica del operador.

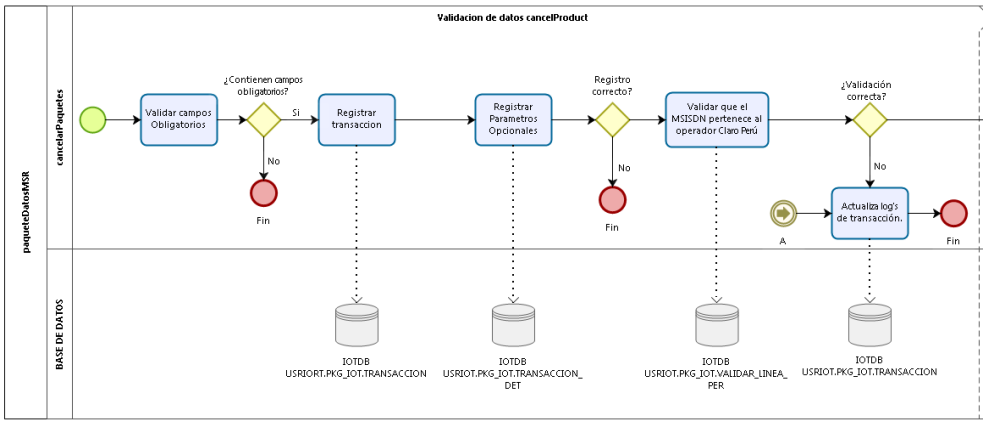

Figura N° 22: Cancelar paquetes (Parte1)
Fuente: Elaboración propia

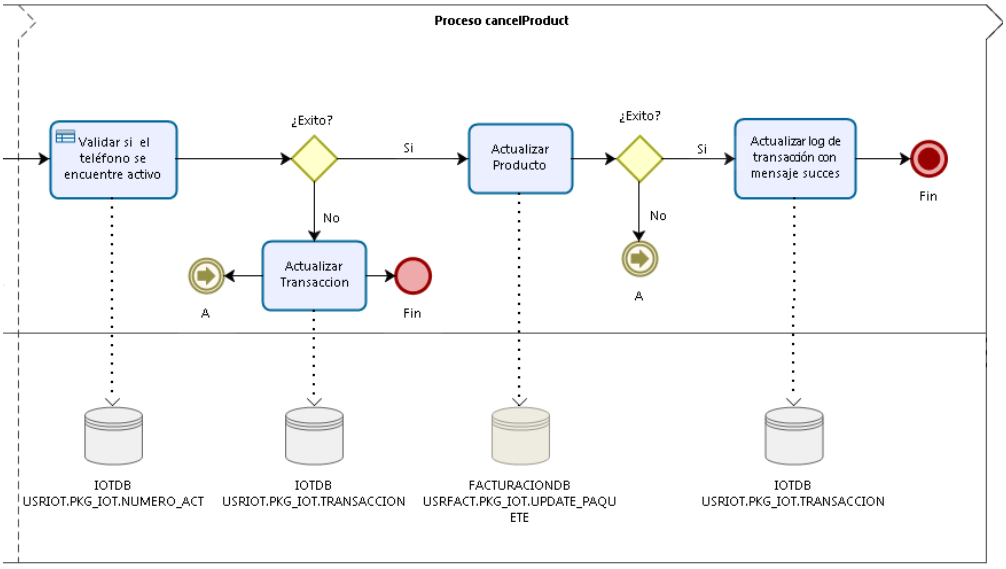

Figura N° 23: Cancelar paquetes (Parte2)
Fuente: Elaboración propia

Gestión Cambio de SIM

Cambiar SIM: Esta operación define las actividades dentro del proceso Gestión Cambio de SIM, para realizar el cambio de SIM de una línea activa, por parte de la empresa de automóviles o del usuario final; desde la plataforma de la empresa de automóviles, hacia la infraestructura tecnológica del operador.

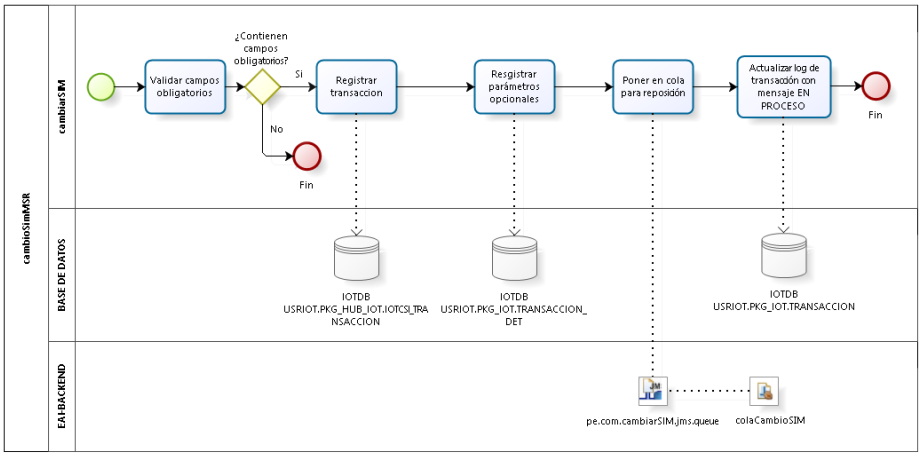

Figura N° 24: Cambio de SIM
Fuente: Elaboración propia

4.1.3. Arquitectura de Aplicación

La arquitectura de aplicación propuesta, provee la definición funcional, para cada uno de los sistemas de información requeridos, y muestra las interacciones entre componentes del sistema con

relación a los procesos clave del negocio. Lo cual, cumple con la siguiente arquitectura:

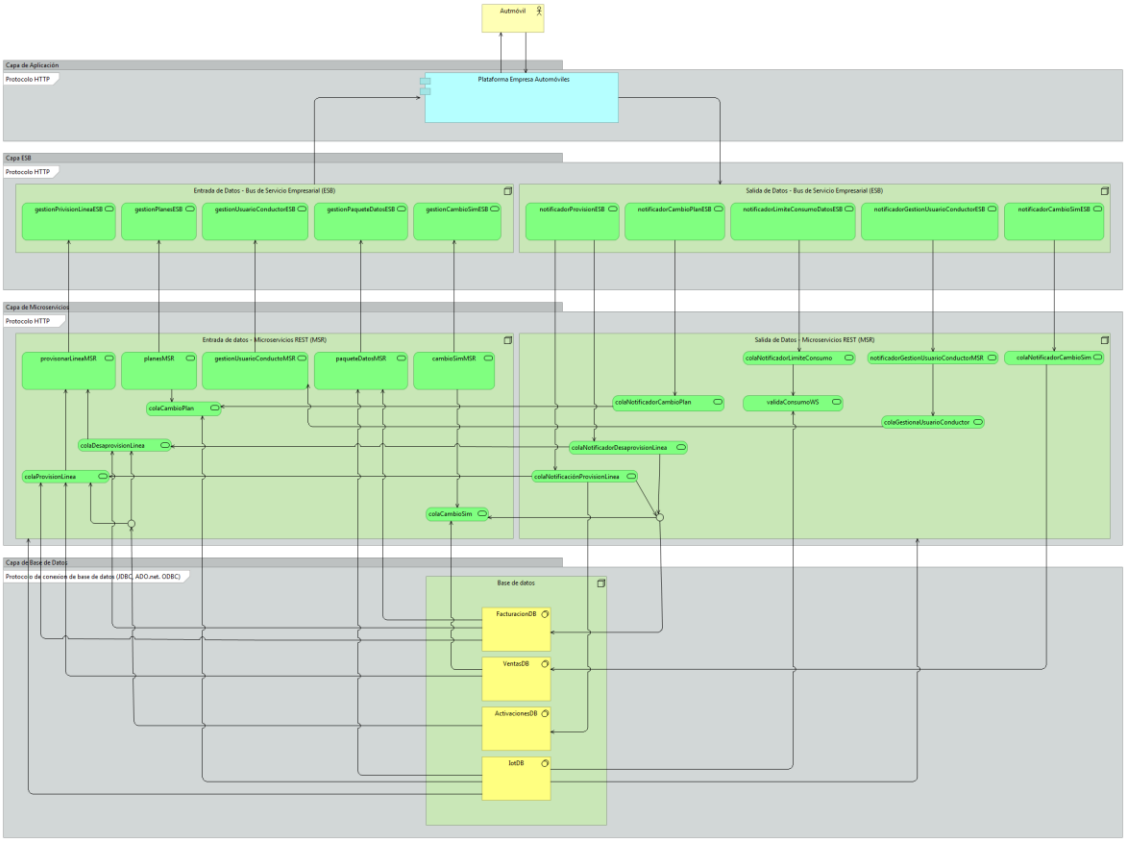

Figura N° 25: Diagrama de Arquitectura de Aplicaciones
Fuente: Elaboración propia

Como se puede apreciar en la Figura N° 25 se diagrama a nivel lógico, la interacción entre los componentes del sistema, segmentados en por cuatros capas:

Capa de Aplicación: Está capa grafica el componente de aplicación desacoplada a la solución del sistema, y es externo a los otros componentes que implementa la arquitectura de aplicación. La comunicación con los componentes ESB es a través del protocolo HTTP, mostrando a un alto nivel, como será la configuración en una implementación física. El componente de aplicación no invocará directamente a los componentes de microservicios, sino, será a través de un bus de comunicaciones, abstrayendo la forma de comunicarse con capas internas de los sistemas de información del negocio.

Capa ESB: Esta capa contiene al bus de comunicación. El bus de comunicación muestra a los componentes implementados tanto en el flujo de entrada, como de salida. Esto permitirá exponer los métodos u operaciones de los microservicios en forma virtual, es decir, el BUS se comunicará con la ruta real del microservicio, pero

expondrá una ruta virtual para el lado del cliente; en este caso, la capa de aplicación. La comunicación también se realizará, a través del protocolo HTTP.

Capa de Microservicios: Esta capa contiene a los componentes de microservicios, que son los que implementan las operaciones de negocios, siendo así, cada microservicio recibirá una petición de la capa de aplicación, a través de un componente en el bus de comunicaciones. En el caso del flujo de salida, el componente de microservicio enviará una respuesta al cliente, a través del mismo bus de comunicaciones con su correspondiente componente. El protocolo de comunicación es HTTP

Capa de Base de Datos: Esta capa contiene la lógica de acceso a la base de datos, el cual permitirá a las operaciones del negocio implementadas en componentes de microservicios, acceder a la información persistente de información. Los protocolos de comunicación pueden ser JDBC, ADO.net, ODBC, etc. Para el presente caso se utilizará JDBC como manejador de conexión de acceso a datos.

4.1.4. Arquitectura Tecnológica

La arquitectura tecnológica propuesta, para cubrir los requerimientos de eficiencia operativa del negocio en clientes ligeros como conceptos de IoT, se describe en el siguiente diagrama:

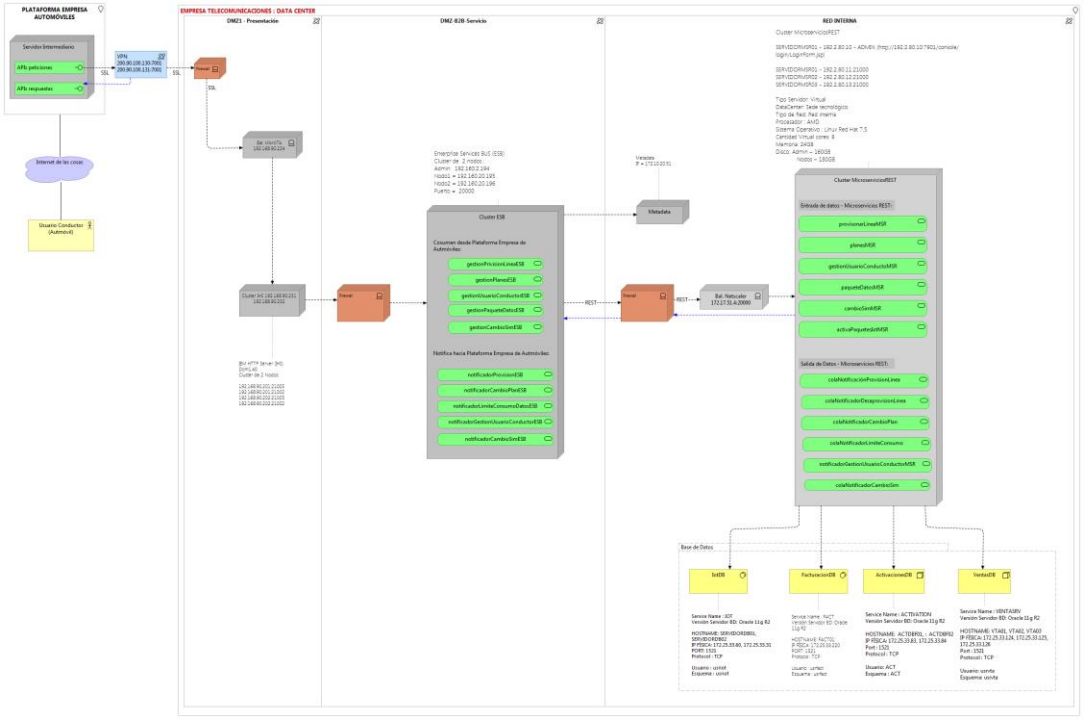

Figura N° 26: Diagrama de Arquitectura Tecnológica
Fuente: Elaboración propia

Como se puede apreciar en la Figura N° 26 se diagrama a nivel físico, la estructura de hardware, software y comunicaciones, sobre la propuesta tecnológica, para dar soporte a los sistemas de información:

Plataforma de Empresa de Automóviles: Esta plataforma es propio del cliente, lo cual debe alinear su infraestructura a la solución tecnológica que ofrece el operador, en este caso, la empresa de telecomunicaciones, que brindará el servicio de comunicación, para los fines comerciales estratégicos del cliente. Este debe contar con todos los requisitos previos antes de implementar los componentes de hardware, software y de comunicaciones.

Data Center: Es la infraestructura tecnológica del operador. La infraestructura está segmentada a nivel de infraestructura y de red por capas:

- DMZ1 – Presentación
    - Nivel de red o comunicaciones se tienen los acceso privado a través de una VPN, con lo cual pasa por un

componente Firewall con conexión SSL, y se comunica con Router. Se enruta con un Servidor HTTP y se comunica con la capa DMZ-B2B de servicio. Esta es el segmento de cara al cliente que es la empresa de automóviles.

- DMZ-B2B-Servicio
    - Este segmento también implementa físicamente un Firewall, lo cual filtra y controla las entradas y salidas de comunicación entre el segmento DMZ1. En este segmento se tiene implementado un BUS de comunicaciones con un clúster de alta disponibilidad de dos nodos y un administrador, es decir, tres servidores físicos configurados, para la implementar los componentes ESB. El BUS de comunicaciones requiere una base de datos para los metadatos.

- Red Interna
    - Este segmento implementa un servidor de base de datos para los metadatos del BUS de comunicaciones. También ser tiene considerado la implementación de un Firewall de red físico. El Firewall filtra y controla las entradas y salidas de comunicación, con el segmento DMZ-B2B-Servicio, y se comunica con un balanceador de carga, para las solicitudes de los microservicios, configurados en un clúster con nombre MicroserviciosREST. Este clúster tiene dos nodos y un administrador en donde se ha de implementar los componentes de microservicios, que tienen la lógica operativa y se comunican con la base de datos del negocio.

- Base de Datos: Infraestructura que se encuentra en la misma red interna de los componentes de microservicios, y provee la información del negocio. En esta capa se implementarán físicamente la estructura de datos de IOT, Facturación, Activaciones y Ventas.

La representación de la transferencia de datos que se utilizará en esta arquitectura será REST en formato JSON. Esto es, para reducir eficientemente el tamaño de los mensajes, y mejorar la rapidez del servicio. Además, los clientes consumidores de los servicios no se verán afectados en el consumo de sus datos, y

tendrán una respuesta más óptima en sus operaciones. Siendo los dispositivos de los automóviles clientes muy ligeros.

4.1.5. Factibilidad Económica

Se ha elaborado un flujo de caja de la empresa analizada, para conocer la viabilidad económica del proyecto, lo cual se muestra en la siguiente Tabla N° 1:

Tabla N° 1: Flujo de Caja
Fuente: Elaboración propia

AÑO	2019	2020	2021	2022
I. MÓDULO DE INVERSIÓN (Expresados en "negativo")	0	-25,000	10,000	11,500
EQUIPOS		25,000	10,000	11,500.00
II. MÓDULO DE OPERACIÓN (A -B)	-252,862	206,161	243,309	308,158
A. INGRESOS INCREMENTALES (a - b)	-372,862	254,461	214,909	282,758
(a) Ingresos con proyecto	0	740,083	814,091	895,500
SERVICIO DE PLANES IOT		740,083	814,090.75	895,499.83
(b) Ingresos sin proyecto	372,862	485,622	599,182	612,742
SERVICIOS DE INTERNET	372,862.00	485,622.00	599,182.00	612,742.00
B. EGRESOS OPERATIVOS INCREMENTALES (c - d)	-120,000	48,300	-28,400	-25,400
(c) Costos y gastos operativos con proyecto	0	180,300	116,800	134,320
CONSULTORIAS		100,300.00	15,300.00	17,595.00
SOPORTE		80,000.00	101,500.00	116,725.00
(d) Costos y gastos operativos sin proyecto	120,000	132,000	145,200	159,720
SOPORTE	50,000.00	55,000.00	60,500.00	66,550.00
EQUIPOS	30,000.00	33,000.00	36,300.00	39,930.00
CONSULTORIAS	40,000.00	44,000.00	48,400.00	53,240.00
FLUJO DE CAJA NOMINAL (I + II)	-252,862	181,161	253,309	319,658
FLUJO DE CAJA ACUMULADO	-252,862	-71,702	181,607	501,265
VALOR ACTUAL NETO (VAN)	266,423.4			
TASA INTERNA DE RETORNO (TIR)	34.54%			
TASA DE DESCUENTO	**15%**			

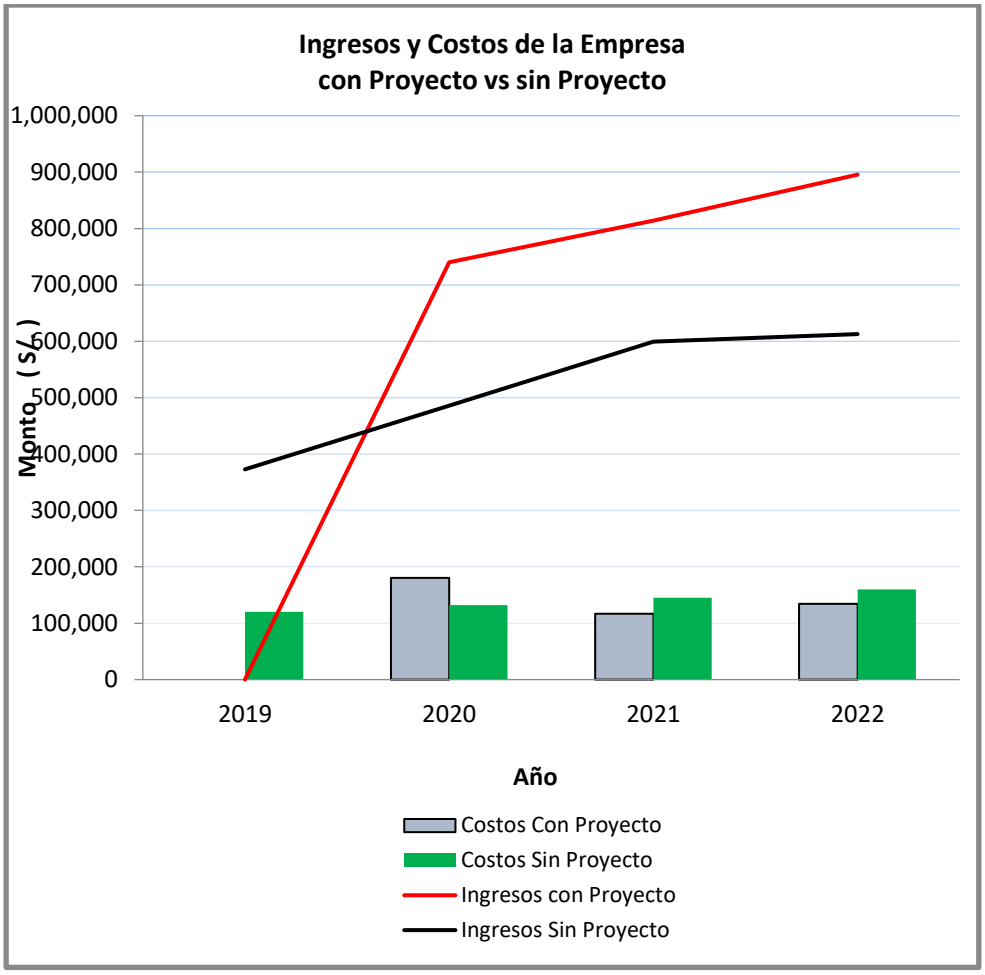

Figura N° 27: Gráfico de Ingresos y Costos de la Empresa con Proyecto vs sin Proyecto

Fuente: Elaboración propia

CAPITULO IV: RECURSOS Y CRONOGRAMA

5.1. Recursos

Se considera al equipo de desarrollo en la siguiente Tabla N° 2:

Tabla N° 2: Actividades y costos
Fuente: Elaboración propia

Actividades	Costo en (S/.)
Recursos Humanos	**21000**
Líder de Proyecto	6000
Analista Funcional (AF)	4000
Arquitecto de Software (ARQ)	4500
Programador de Base de Datos (AP1-BD)	3000
Programador SOA (AP2-EAI)	3500
Viáticos y servicios	**300**
Gastos de movilidad.	300
TOTAL	**S/. 21,300**

5.2. Cronograma de ejecución

Se estima la duración del proyecto en 61 días aproximadamente, considerando el calendario del proyecto de lunes a viernes, sin feriados. El proyecto iniciará el 6 de enero 2020 y culminará el 27 de marzo de 2020.

El equipo de desarrollo del proyecto estará conformado por los recursos señalados en la Tabla N°2.

Nombre de tarea	Duración	Trabajo	Comienzo	Fin
Proyecto IOT - Microservicios	60.13 días	1,353.6 horas	lun 06/01/20	vie 27/03/20
Gestión del Proyecto	57 días	45.6 horas	lun 06/01/20	mar 24/03/20
Seguimiento Técnico del Proyecto	57 días	45.6 horas	lun 06/01/20	mar 24/03/20
Analisis, Diseño y Desarrollo	51 días	780 horas	lun 06/01/20	lun 16/03/20
Análisis y Arquitectura	22 días	344 horas	lun 06/01/20	mar 04/02/20
ANÁLISIS	16 días	168 horas	lun 06/01/20	lun 27/01/20
DISEÑO	17 días	176 horas	lun 13/01/20	mar 04/02/20
DESARROLLO	29 días	436 horas	mié 05/02/20	lun 16/03/20
Revisión de información para pruebas unitarias y funcionales	1 día	32 horas	mié 05/02/20	mié 05/02/20
OBJETOS DE BASE DE DATOS DISEÑADOS	0.5 días	4 horas	jue 06/02/20	jue 06/02/20
COMPONENTES EAI DISEÑADOS	10 días	80 horas	jue 06/02/20	mié 19/02/20
PRUEBAS	11.5 días	96 horas	jue 06/02/20	vie 21/02/20
DOCUMENTACIÓN	17 días	224 horas	vie 21/02/20	lun 16/03/20
CERTIFICACIÓN	22.5 días	508 horas	lun 24/02/20	mié 25/03/20
PRUEBAS CICLO 1	10 días	240 horas	lun 24/02/20	vie 06/03/20
PRUEBAS CICLO 2	4 días	96 horas	lun 09/03/20	jue 12/03/20
CIERRE DE CERTIFICACIÓN	1.5 días	36 horas	vie 13/03/20	lun 16/03/20
SUSTENTACIÓN A COMITÉ REALIZADA	0.5 días	12 horas	lun 16/03/20	lun 16/03/20
PASE A PRODUCCIÓN REALIZADA	1.5 días	36 horas	mar 17/03/20	mié 18/03/20
MONITOREO DE PASE A PRODUCCIÓN REALIZADA	5 días	88 horas	mié 18/03/20	mié 25/03/20
Validar y Estabilizar el Sistema en Producción	1 día	24 horas	mié 18/03/20	jue 19/03/20
Soporte post Producción	4 días	64 horas	jue 19/03/20	mié 25/03/20
CIERRE DEL PROYECTO	2.5 días	20 horas	mié 25/03/20	vie 27/03/20
Gestionar el Cierre Administrativo del Proyecto	1.5 días	12 horas	mié 25/03/20	jue 26/03/20
Elaborar acta de cierre	0.5 días	4 horas	vie 27/03/20	vie 27/03/20
Acta de cierre firmada	0.5 días	4 horas	vie 27/03/20	vie 27/03/20

Figura N° 28: Cronograma de actividades del proyecto
Fuente: Elaboración propia

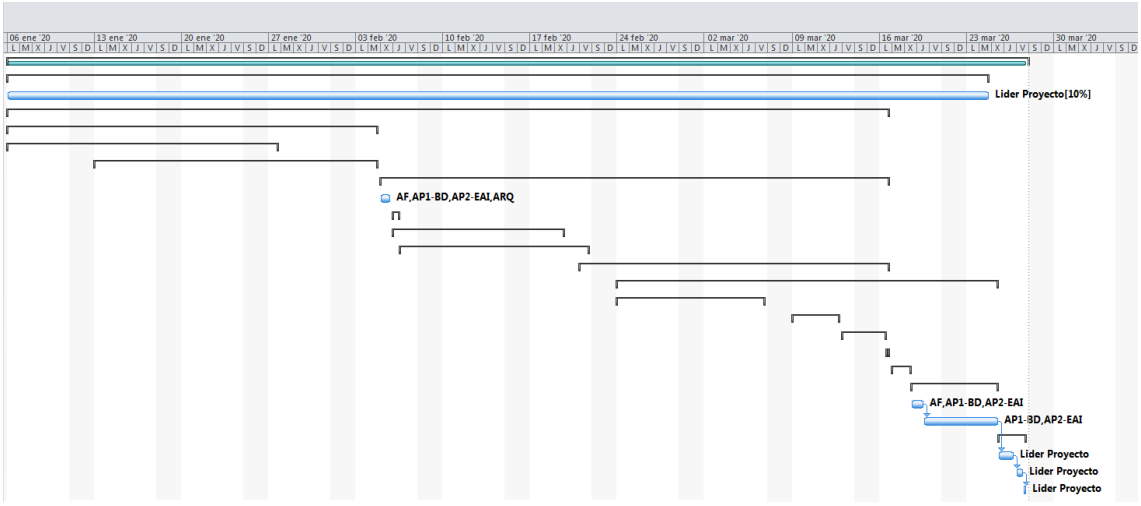

Figura N° 29: Gantt de actividades del proyecto
Fuente: Elaboración propia

CAPITULO V: CONCLUSIONES Y RECOMENDACIONES

6.1. Conclusiones

1. Determinar la relación que existe, entre la Arquitectura tecnológica como parte de la Arquitectura de Microservicios (MSA), y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú.

   Por lo tanto, se ha demostrado a nivel de la arquitectura de aplicaciones y tecnológica, la implementación de una arquitectura de microservicios en el sistema, permite mayor flexibilidad de escalabilidad en el control de cambios de los servicios, segmentando la funcionalidad en componentes independientes, para realizar una operación específica dentro del proceso de negocio. Se considera un también un bus de comunicaciones en la arquitectura propuesta, para transformar, enrutar, abstraer y encapsular la comunicación entre la aplicación y los componentes de microservicios, permitiendo la comunicación entre los componentes del sistema, utilizar cualquier protocolo de comunicación.

2. Determinar la relación que existe, entre la Arquitectura REST como estandarización en la Arquitectura de Microservicios (MSA), y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú.

   Por lo tanto, se ha demostrado a nivel de la arquitectura de aplicaciones y tecnológica, que la implementación de una especificación REST como estándar de microservicios en el sistema, permite mayor rapidez y eficiencia en las operaciones en los procesos del negocio. Esto porque REST se apoya concretamente en las especificaciones de HTTP y es una alternativa a la arquitectura SOAP, permitiendo utilizar al cien por ciento, todos los recursos de comunicación que se obtienen a través del protocolo HTTP, manipulando los recursos a través de las URI.

   Con REST es más fácil dividir los componentes de software en microservicios, por ser independiente de la plataforma permitiendo la portabilidad en distintos servidores de distintos lenguajes, lo que permite desacoplarse totalmente entre la capa de aplicación, como la capa de base de datos.

   En REST, los intercambios de mensajes entre componentes, comúnmente se pueden utilizar los formatos XML y JSON, al contrario de SOAP que solo trabaja con XML.

6.2. Recomendaciones

Si bien una solución de sistema de información basada en una arquitectura de Microservicios se logra mayor escalabilidad, operatividad y desacoplamiento; no está ajeno a las vulnerabilidades del sistema. Un sistema monolítico puede ser menos vulnerable, pero se sacrifica la eficiencia operativa.

Para garantizar la seguridad y no perder la eficiencia en la operatividad, y así poder trabajar con el esquema de Microservicios, se recomienda una implementación de hardware, software y comunicaciones de acuerdo a la arquitectura tecnológica propuesta.

Otro factor que se recomienda, es trabajar sobre la especificación REST, siendo el de intercambio de datos en formato JSON; ya que este tipo de formato o estructura tiene ventajas sobre el formato XML, por ser más ligero, sin mucho metadato en su estructura.

CAPITULO VI: FUENTES DE INFORMACION

7.1. Referencias bibliográficas

Newman, S. (2015). Building Microservices DESIGNING FINE-GRAINED SYSTEMS. United States of America: O'Reilly Media, Inc.

Maya, E., & López, D. (2018). Arquitectura de Software basada en Microservicios para Desarrollo de Aplicaciones Web. Obtenido de https://www.researchgate.net/publication/328887426_Arquitectura_de_Software_basada_en_Microservicios_para_Desarrollo_de_Aplicaciones_Web

Arévalo del río, Francisco Vicente (2017). EL ENFOQUE DE MICROSERVICIOS COMO ESTRATEGIA PARA MEJORAR LA CALIDAD DEL SOFWARE.

Cayo Alcos, Diego Benito (2018). EH-UNMSM: E-Health Cloud para la mejora de procesos de la clínica universitaria UNMSM mediante una arquitectura de microservices.

Arboleda Cola, Carlos Augusto (2017). Propuesta metodológica para migración de sistemas web con arquitectura monolítica hacia una arquitectura basada en microservicios.

Velepucha, Víctor; Flores, Pamela & Torres, Jenny (2019). MOMMIV: Modelo para descomposición de una arquitectura monolítica hacia una arquitectura de microservicios bajo el Principio de Ocultación de Información.

Saransig Chiza, Alexis Fernando (2018). Análisis de rendimiento entre una arquitectura monolítica y una arquitectura de microservicios – tecnología basada en contenedores.

Ruelas Acero, Donia Alizandra (2017). Modelo de composición de microservicios para la implementación de una aplicación web de comercio electrónico utilizando kubernetes.

Karen Tatiana Gómez Suárez, Raquel Anaya & Andrés Felipe Cano (2017).Un acercamiento a los microservicios.

Cabrera Alvarado, E. F., & Cárdenas Cárdenas, P. J. (2018-10-29). Tesis. Recuperado a partir de http://dspace.ucuenca.edu.ec/handle/123456789/31511

Nebel, A. (2019.). Arquitectura de microservicios para plataformas de integración. Tesis de maestría. Universidad de la República (Uruguay). Facultad de Ingeniería.

7.2. Referencias electrónicas

Lewis, J., & Fowler, M. (25 de Marzo de 2014). *Microservices*. Obtenido de martinfowler.com: https://martinfowler.com/articles/microservices.html

(Pereira & Rumiche 2017). Migración de los servicios de pagos en línea para soportar transacciones en el aplicativo móvil de una empresa de telecomunicaciones http://www.repositorioacademico.usmp.edu.pe/handle/usmp/3783

https://www.gartner.com/en/documents/3898774/how-to-succeed-with-microservices-architecture-using-dev

https://es.wikipedia.org/wiki/Arquitectura_orientada_a_servicios

https://www.ibm.com/cloud/learn/esb

https://developer.mozilla.org/es/docs/Glossary/REST

https://www.json.org/json-en.html

https://www.redhat.com/es/topics/microservices

https://www2.deloitte.com/es/es/pages/technology/articles/IoT-internet-of-things.html

https://www.redhat.com/es/topics/devops

https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces

https://es.wikipedia.org/wiki/MSISDN

https://es.wikipedia.org/wiki/Tarjeta_SIM

https://es.wikipedia.org/wiki/IMSI

https://www.cisco.com/c/en/us/products/security/firewalls/what-is-a-firewall.html

https://www.linksys.com/sv/support-article?articleNum=142514

https://es.wikipedia.org/wiki/Router

https://es.wikipedia.org/wiki/Cl%C3%BAster_(inform%C3%A1tica)

https://es.wikipedia.org/wiki/Balanceador_de_carga

ANEXOS

FORMULACIÓN DEL PROBLEMA	OBJETIVOS	HIPÓTESIS	VARIABLE
Problema general.-	Objetivo General.-	Hipótesis general.-	V. Independiente
¿Cuál es la relación que existe entre la Arquitectura de Microservicios y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú?	Determinar la relación que existe entre Arquitectura de Microservicios y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú.	Existe una relación significativa entre la Arquitectura de Microservicios y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú	Arquitectura de Microservicios (MSA).
Problemas Específicos.-	Objetivos Específicos.-	Hipótesis Especificas.-	V. Dependiente
1. ¿Cuál es la relación que existe, entre la Arquitectura tecnológica, como parte de la Arquitectura de Microservicios (MSA), y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú? 2. ¿Cuál es la relación que existe, entre la Arquitectura REST como estandarización en la Arquitectura de Microservicios (MSA), y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú?	1. Determinar la relación que existe, entre la Arquitectura tecnológica como parte de la Arquitectura de Microservicios (MSA), y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú. 2. Determinar la relación que existe, entre la Arquitectura REST como estandarización en la Arquitectura de Microservicios (MSA), y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú.	1. Existe una relación significativa, entre la Arquitectura tecnológica como parte de la Arquitectura de Microservicios (MSA), y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú. 2. Existe una relación significativa entre la Arquitectura REST como estandarización en la Arquitectura de Microservicios (MSA), y la eficiencia operativa en los procesos de negocio de dispositivos IOT, para una empresa de telecomunicaciones en el Perú.	Eficiencia operativa en los procesos de negocio de dispositivos IOT.